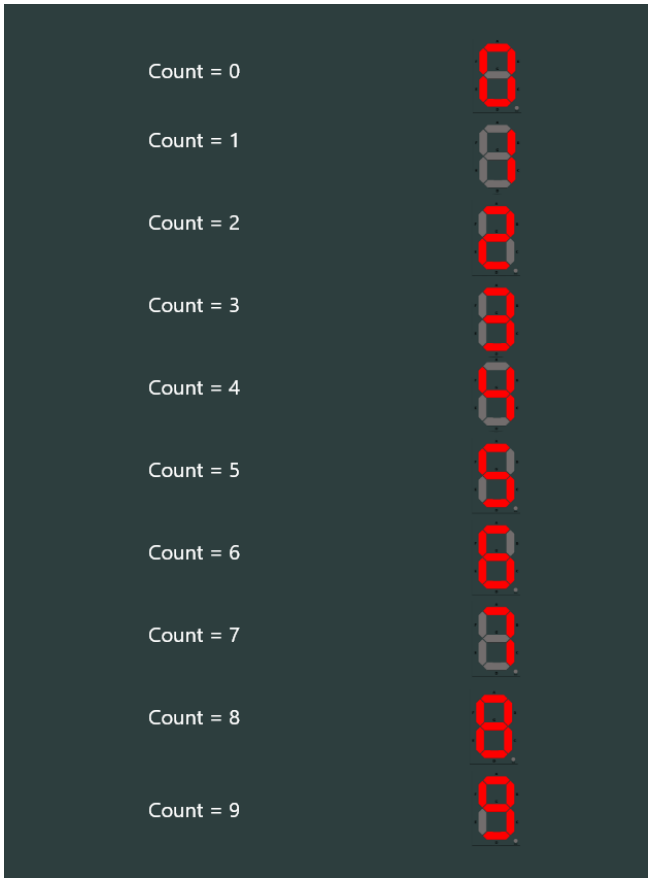
7-segment Counter with STM32F0

Seven-segment display is the most common device used for displaying digits and the alphabet. You can see the Seven Segment Display devices in TV shows counting down to ‘0’. The use of LEDs in seven-segment displays made it more popular.

The binary information can be displayed in the form of decimals using this seven-segment display. Its wide range of applications is in microwave ovens, calculators, washing machines, radios, digital clocks etc.

Tutorial, we will make 7-segment counters that show 0-9 digits when pressing the push button one by one. When the push button is pressed it increments the count value by one and the 7-segment value also increments.



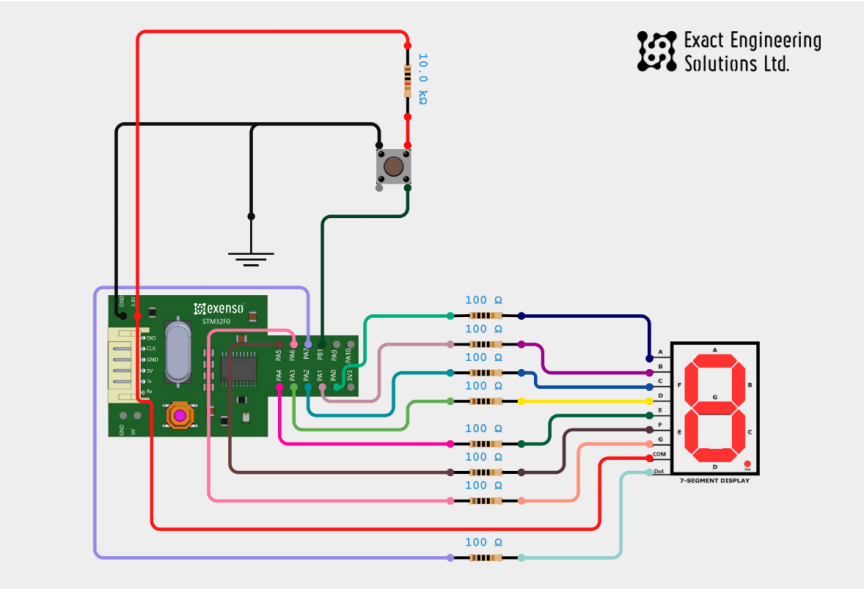
Components Required

You will need the following components −

- 1 × Breadboard
- 1 × STM32F030F4P6
- 1× 7-segment Display
- 1x push button
- 8 × 100Ω Resistor
- 1 × 10kΩ Resistor
- Some Jumper wire

Procedure

Follow the circuit diagram and hook up the components on the breadboard as shown in the image given below.

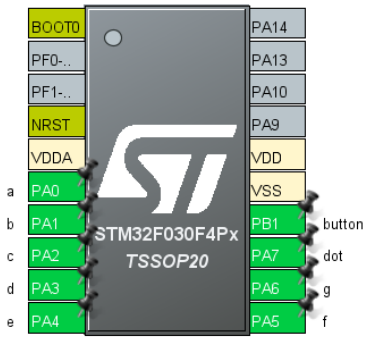


- The PB1 pin will read the output of the push button, if the state is active low, it will increment the 7segment values. Because the push button is connected with the PB1 pin in pull-up configuration and under normal conditions active high output will appear on the PB1 pin

- When the push button is pressed, the count value increments by 1, and Seven-Segment shows the output value.

STM32F0 Pin Configuration:

PA0 to PA7 pin outputs and PB1 pin Input.



CODE

```
#include "main.h"

void SystemClock_Config(void);
static void MX_GPIO_Init(void);
uint8_t seg_table[] = {

      0b11000000,    // 0
      0b11111001,    // 1
      0b10100100,    // 2
      0b10110000,    // 3
      0b10011001,    // 4
      0b10010010,    // 5
      0b10000010,    // 6
      0b11111000,    // 7
      0b10000000,    // 8
      0b10010000     // 9
};

int length = sizeof(seg_table) / sizeof(seg_table[0]);

int main(void)
{
  HAL_Init();

  SystemClock_Config();

  MX_GPIO_Init();

  uint8_t button,button_os;
  uint8_t count = 0;

  button_os = HAL_GPIO_ReadPin(button_GPIO_Port, button_Pin);

  while (1)
  {
        button = HAL_GPIO_ReadPin(button_GPIO_Port, button_Pin);

        if(button != button_os){

            HAL_Delay(10);             // hold 10ms
            button_os = button;

            if(button == GPIO_PIN_RESET){
                count ++;

                if(count == length){
                    count =0;
                  }
              }
        }

        GPIOA->ODR = seg_table[count];   //ODR-Output Data Register value

  }

}
```